NYU Tandon School of Engineering
Computer Science and Engineering
CS6083, Spring 2023

December 2, 2023

Project #2 (due around December 18)

In the second part of the project, you need to create a web-based user interface for the database designed in the first project. In particular, the interface should allow users to register, to login, and to list, add, and remove service locations and smart devices. When adding a new smart device, the system should allow the user to select the type of device (refrigerator, AC, dryer, etc), and then to choose from a list of prestored models (Bosch 800 refrigerator, IKEA smart light bulb, GE 4500 Air Conditioner, etc.) for that type of device.

In addition, users should be able to access a few (say 4-5) different views of their energy consumption pattern. For example, one view might be to simply show the daily energy consumption during a selected time period — basically a graph with time on the x-axis, and the energy consumption on a given day (or week, or month, depending on resolution of the graph) on the y-axis. Another view might show energy consumption per device for a given time period, say the last month, so people can figure out which devices use how much energy. Or you could show how their energy consumption during a time period compares to the average consumption of other locations that are similar (meaning, similar square footage and number of residents). Or you could have a view that also considers prices and maybe shows when devices were used at peak (high-price) times and how much could have been saved by using them during a different time (say by running the dryer overnight when energy is cheaper). These are just suggestions — you can implement any 4-5 different charts and other representations that you think might be useful for understanding energy consumption and cost. Note that **you are not expected** to implement any payment- and billing-related features such as current balance, payments, or auto-pay, which are beyond the scope of this project.

Note that you have more freedom in this second project to design your own system. You still have to follow the basic guidelines, but you can choose the actual look and feel of the site and the energy consumption views and charts that you want to implement, and offer other features that you find useful. In general, design an overall nice and functional system. If you are doing the project as a group of two, note that both students have to attend the demo and know ALL details of the design. So work together with your partner, not separately. Start by revising your design from the first project as needed to best support the system.

Users should be able to perform all operations via a standard web browser. Your system does not have to be available on the public internet. Instead, the most common setup is to install a web server on your laptop, and to then use your web browser on the same laptop to access that web server locally. You can use many frameworks for implementing this, including PHP,

Python Flask, Django, etc. However, you should still have to write SQL queries for your system that you can show us during the demo— so don't go all out on exploiting the ORM features of some of the frameworks. Contact the TAs for technical questions. The main restriction is that the backend should be a relational DBMS with the schema you designed in the first part, with suitable adjustments as needed.

For full credit, your interface should take appropriate measures to guard against SQL injection and cross-site scripting attacks, and to support concurrent access. To prevent SQL injection, you may use stored procedures and prepared statements (if your programming language supports them). If your language does not support prepared statements, your code should check and sanitize inputs from users before concatenating them into query strings. To guard against cross-site scripting, outputs to be returned to user's browsers should be checked or sanitized to avoid scripts. Some languages provide functions, such as htmlspecialchars in PHP, to help with this. You should also define appropriate transactions to make sure that multiple users can use the site at the same time. Make sure to address these requirements (protection against SQL injections etc., and concurrency) in your final document, and to describe your solutions. You may also be asked about this during the demo, and be expected to show and explain what you did.

Every group is expected to demo their project for about 15 minutes to one of the TAs at the end of the semester. **There will be several days from which you can choose for your demo, including days before and after the final.** Your submission is due before the demo. If you use your own installation, make sure you can access this during the demo. Grading will be done on the entire project based on what features are supported, how convenient the system is, your project description and documentation (important!), and the appropriateness of your design in terms of overall architecture and use of the DBMS. Make sure to input some interesting data so you can give a good demo.

Describe and document your design. Log some sessions with your system. You should also be able to show and explain your source code during the demo. The documentation should consist of 15 to 20 pages of carefully written text and suitable figures, describing and justifying your design and the decisions you made during the implementation, and describing how a user should use your system. Note that your documentation and other materials should cover both Projects 1 and 2, so you should modify and extend your materials from the first project appropriately. There will be opportunity to get extra credit by implementing cool extra features, but extra credit is limited to about 5-10% and the TAs will decide what is cool. There might also be extra extra credit of up to 3% for doing an early demo before the deadline.